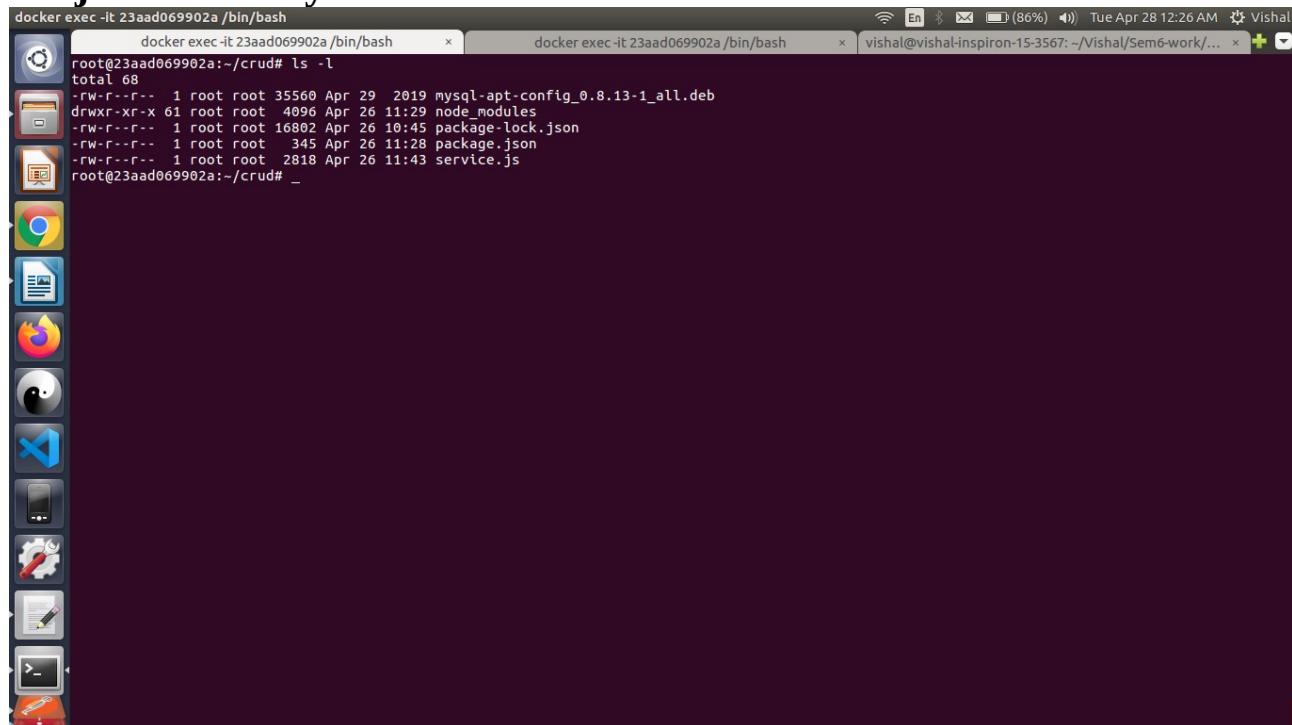


Roll Number:17134
Name: Vishal Kokane
Assignment 2:Docker
Git URL:<https://github.com/vishalkokane264/pucsd2020-devops>

This assignment is of building REST api using CRUD operations.
I've used nodejs +mysql in base docker image of ubuntu.

Project Directory structure:



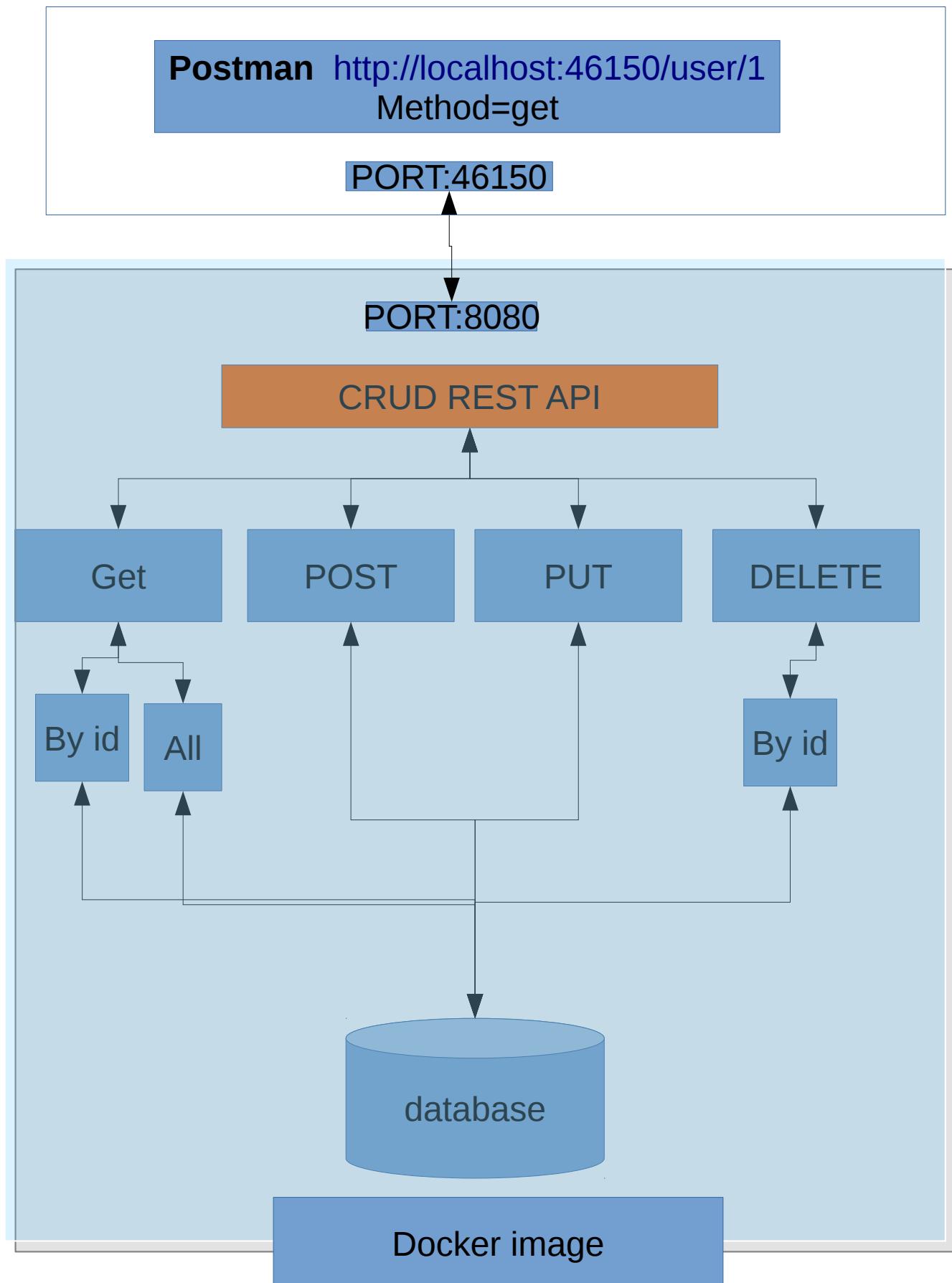
A screenshot of a Linux desktop environment. On the left is a dock with various icons for applications like a terminal, file manager, browser, and code editor. The main area shows two terminal windows. The first window is titled 'docker exec -it 23aad069902a /bin/bash' and contains the command 'ls -l' followed by a listing of files in the current directory. The second window is titled 'vishal@vishal-inspiron-15-3567: ~/Vishal/Sem6-work...' and shows the user's home directory. The desktop background is dark.

```
root@23aad069902a:~/crud# ls -l
total 68
-rw-r--r-- 1 root root 35560 Apr 29 2019 mysql-apt-config_0.8.13-1_all.deb
drwxr-xr-x 61 root root 4096 Apr 26 11:29 node_modules
-rw-r--r-- 1 root root 16802 Apr 26 10:45 package-lock.json
-rw-r--r-- 1 root root 345 Apr 26 11:28 package.json
-rw-r--r-- 1 root root 2818 Apr 26 11:43 service.js
root@23aad069902a:~/crud# _
```

Api Documentation:

Method	URL	Description	
GET by id	http://localhost:46150/user/:id	Get user by user id	
Get (all)	http://localhost:46150/user	Get all users	
POST	http://localhost:46150/user	Add user in database	
PUT	http://localhost:46150/user	Update username using userid	
DELETE(id)	http://localhost:46150/user/id	Delete user using userid	

Flow diagram



dockerfile:

```
FROM debian:latest

RUN cd
RUN apt-get update && apt-get install -y wget
RUN wget https://dev.mysql.com/get/mysql-apt-config_0.8.13-1_all.deb
RUN dpkg -i mysql-apt-config_0.8.13-1_all.deb
RUN apt-get update
RUN apt-get install -y mysql-server && apt-get
install -y nodejs && apt-get install -y npm

COPY nodejs/ /root/crud

RUN cd /root/crud
RUN npm install

RUN /etc/init.d/mysql stop

RUN mysqld_safe --skip-grant-tables &

RUN mysql < database.sql

EXPOSE 8080

CMD [ "node", "service.js" ]
```

Snapshot:

Running Docker file

```
docker exec -it 23aad069902a /bin/bash
  docker exec -it 23aad069902a /bin/bash
  docker exec -it 23aad069902a /bin/bash
vishal@vishal-inspiron-15-3567:~/Vishal/Sem6-work... + Visha

1588 docker build -t "mysql_ubuntu:dockerfile27" .
1589 docker build -t "mysql_ubuntu:dockerfile" .
1603 docker build -t "dockerbuild:dockerfile" .
1607 docker build -t "dockerbuild:dockerfile" .
1614 docker run -d -p 8070:8070 --name dockerbuild:dockerfile -e MYSQL_ROOT_PASSWORD=' dockerbuild:dockerfile
1623 docker build -t "dockerbuild:dockerfile" .
1669 docker build -t "dockerbuild:dockerfile" .
+ project2 git:(master) X docker build -t "crudrest:dockerfile" .
Sending build context to Docker daemon 2.048kB
Step 1/7 : FROM crud2:latest
--> b4928ea4ac53
Step 2/7 : RUN cd
--> Running in 03655e63c121
Removing intermediate container 03655e63c121
--> d63fdb8e8288
Step 3/7 : RUN cd /root/crud2
--> Running in 73eb6783ca28
Removing intermediate container 73eb6783ca28
--> 807f5dca274c
Step 4/7 : RUN /etc/init.d/mysql stop
--> Running in e2d35ff52fd6
[Info] MySQL Community Server 5.7.29 is already stopped.
Removing intermediate container e2d35ff52fd6
--> f14103dd9d64
Step 5/7 : RUN mysqld_safe --skip-grant-tables &
--> Running in 5996fbe7ecc3
Removing intermediate container 5996fbe7ecc3
--> 962be8cb20a8
Step 6/7 : EXPOSE 8080
--> Running in bddef1db282
Removing intermediate container bddef1db282
--> 1865fb7322b
Step 7/7 : CMD ["node","index.js"]
--> Running in d3c3983e4cf6
Removing intermediate container d3c3983e4cf6
--> b58dbc12f077
Successfully built b58dbc12f077
Successfully tagged crudrest:dockerfile
+ project2 git:(master) X docker images
REPOSITORY          TAG      IMAGE ID      CREATED     SIZE
crudrest            dockerfile  b58dbc12f077  14 seconds ago  1.14GB
crud2               latest    b4928ea4ac53   4 minutes ago  1.14GB

vishal@vishal-inspiron-15-3567:~/Vishal/Sem6-work/pucsd2020-Tasks/Training/project2
```

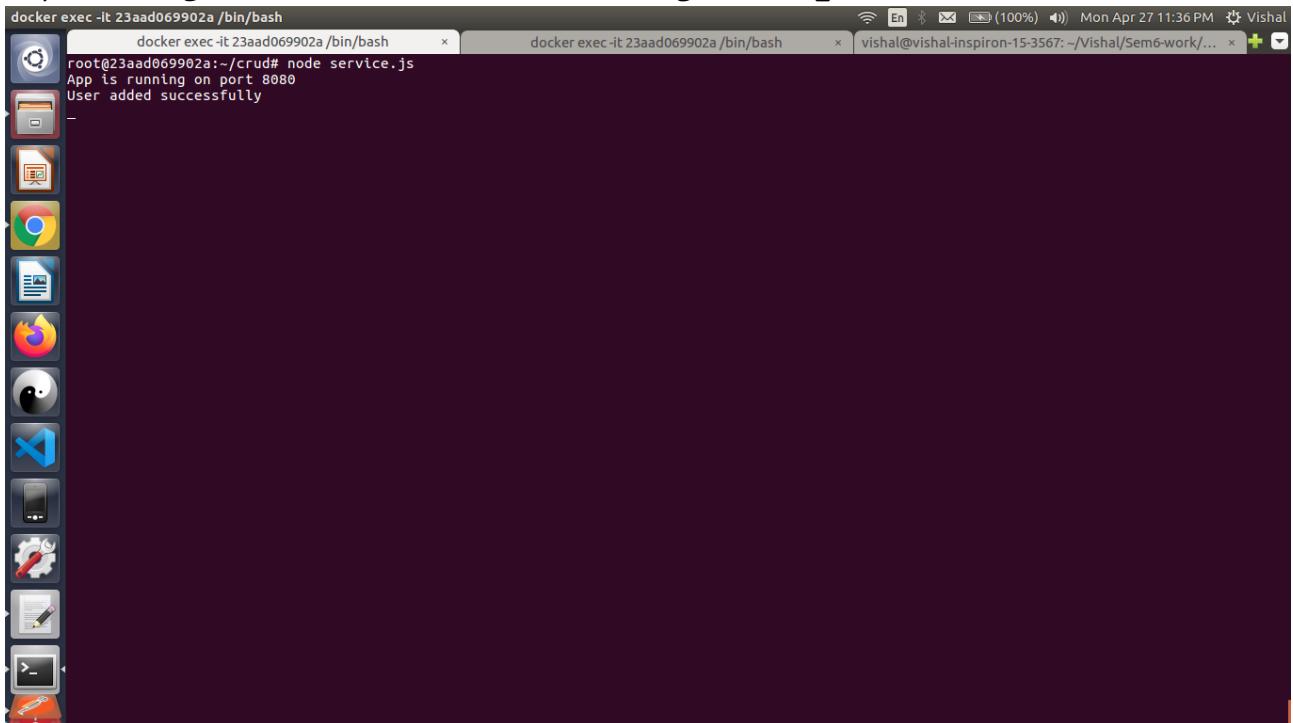
1) Docker images && container running

```
vishal@vishal-inspiron-15-3567:~/Vishal/Sem6-work/pucsd2020-Tasks/Training/project2
  docker exec -it 23aad069902a /bin/bash
  docker exec -it 23aad069902a /bin/bash
  vishal@vishal-inspiron-15-3567:~/Vishal/Sem6-work... + Visha

+ project2 git:(master) X docker images
REPOSITORY          TAG      IMAGE ID      CREATED     SIZE
crudrest            dockerfile  b58dbc12f077  14 minutes ago  1.14GB
crud2               latest    b4928ea4ac53   18 minutes ago  1.14GB
vishal              latest    33ceed915f0c   2 hours ago   1.1GB
helloworld          latest    9dd3633380cc  2 hours ago   1.1GB
dockerbuild         dockerfile  53caa99b262   10 hours ago   1.1GB
new2                latest    a515a462ddd   10 hours ago   1.1GB
mysql-node-ubuntu   latest    a21d9eb750d8   11 hours ago   1.1GB
mysql_ubuntu         dockerfile  f9b78f8c5341  12 hours ago   644MB
crud-service        latest    e9a0cfb89b3d   30 hours ago   1GB
ubuntu              latest    1d622ef86b13   3 days ago    73.9MB
debian              latest    971452c94376  2 months ago   114MB
herreralutis/mysql-ubuntu latest    293d44326d32  3 years ago    345MB

+ project2 git:(master) X docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
23aad069902a        b58dbc12f077   "/bin/bash"        13 minutes ago   Up 13 minutes   0.0.0.0:46150->8080/tcp   crazy_goldwasser
+ project2 git:(master) X _
```

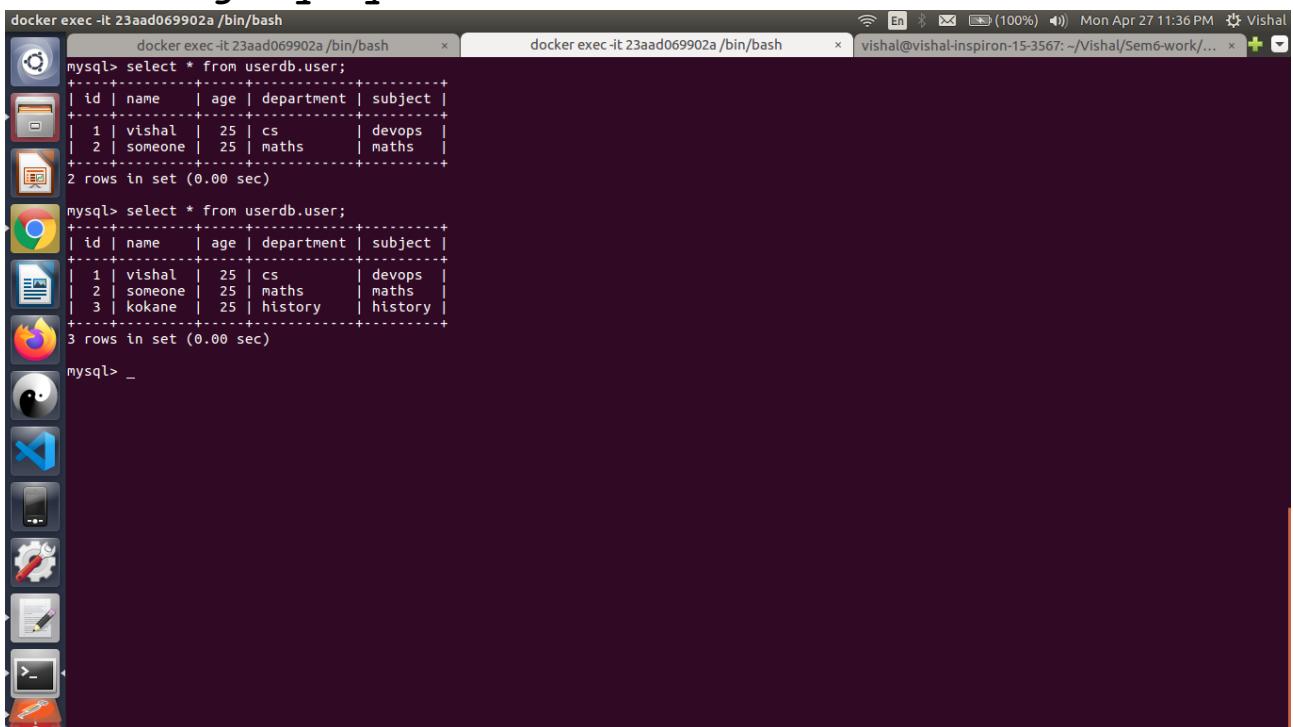
2) Nodejs service running on port 8080



A screenshot of an Ubuntu desktop environment. On the left is a dock with various icons. In the center, there are three terminal windows. The first terminal window shows the command 'docker exec -it 23aad069902a /bin/bash' and the output of a Node.js script running a MongoDB application. The second terminal window shows the command 'docker exec -it 23aad069902a /bin/bash'. The third terminal window shows the command 'vishal@vishal-inspiron-15-3567: ~/Vishal/Sem6-work...'.

```
root@23aad069902a:~/crud# node service.js
App is running on port 8080
User added successfully
```

3) Another same container running for checking mysql data



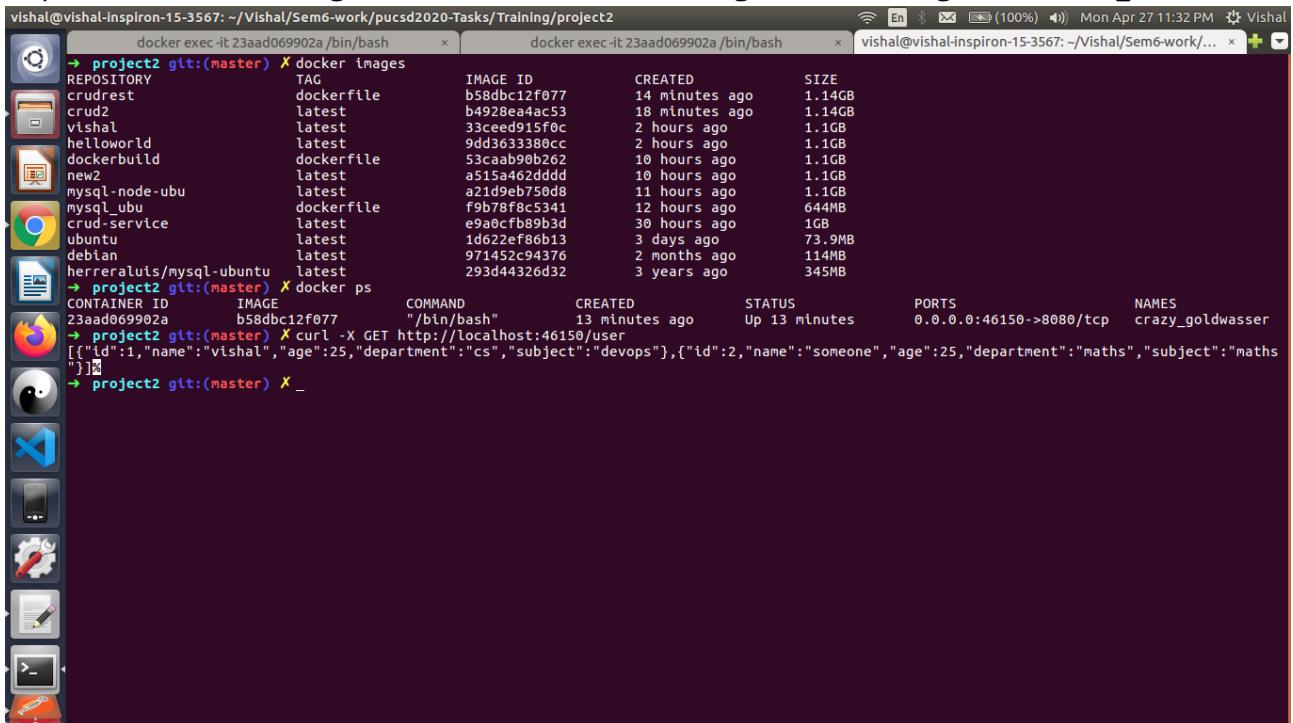
A screenshot of an Ubuntu desktop environment. On the left is a dock with various icons. In the center, there are two terminal windows. The first terminal window shows the command 'docker exec -it 23aad069902a /bin/bash' and MySQL queries against a 'userdb' database. The second terminal window shows the command 'vishal@vishal-inspiron-15-3567: ~/Vishal/Sem6-work...'.

```
mysql> select * from userdb.user;
+----+-----+-----+-----+-----+
| id | name | age | department | subject |
+----+-----+-----+-----+-----+
| 1  | vishal | 25 | cs       | devops   |
| 2  | someone | 25 | maths    | maths    |
+----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> select * from userdb.user;
+----+-----+-----+-----+-----+
| id | name | age | department | subject |
+----+-----+-----+-----+-----+
| 1  | vishal | 25 | cs       | devops   |
| 2  | someone | 25 | maths    | maths    |
| 3  | kokane  | 25 | history  | history  |
+----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> _
```

4) Validating service using CURL get request

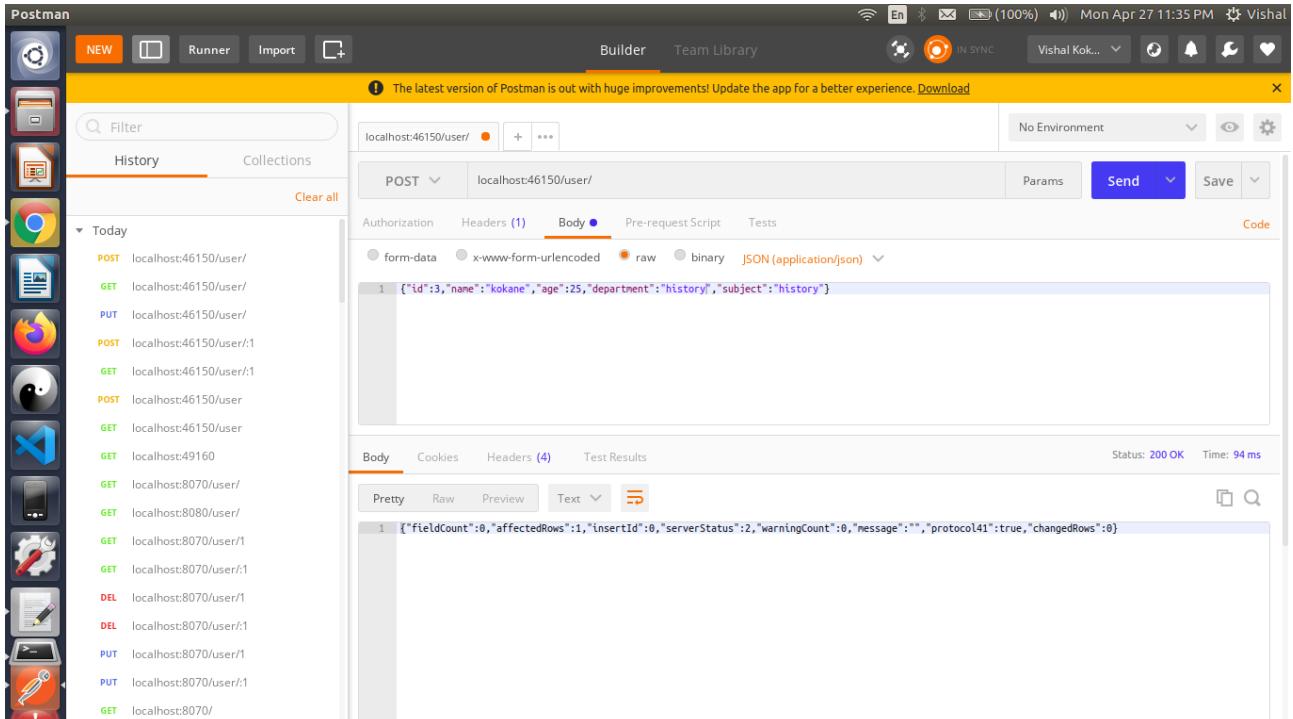


```
vishal@vishal-Inspiron-15-3567: ~/Vishal/Sem6-work/pucsd2020-Tasks/Training/project2
  docker exec -it 23aad069902a /bin/bash
+ project2 git:(master) X docker images
REPOSITORY          TAG        IMAGE ID      CREATED       SIZE
crudrest            latest     b58dbc12f077  14 minutes ago  1.14GB
crud2               latest     b4928ea4ac53  18 minutes ago  1.14GB
vishal              latest     33ceed915f0c  2 hours ago   1.1GB
helloworld          latest     9dd3633380cc  2 hours ago   1.1GB
dockerbuild         dockerfile  53caab90b262  10 hours ago  1.1GB
new2                latest     a515a462ddd   10 hours ago  1.1GB
mysql-node-ubuntu   latest     az1d9eb750d8  11 hours ago  1.1GB
mysql_ubuntu         dockerfile  f9b78f8c5341  12 hours ago  644MB
crud-service         latest     e9a0cfb89b3d  30 hours ago  1GB
ubuntu              latest     1d622ef86b13  3 days ago   73.9MB
debian              latest     971452c94376  2 months ago  114MB
herreraluis/mysql-ubuntu latest   293d44326d32  3 years ago   345MB

+ project2 git:(master) X docker ps
CONTAINER ID        IMAGE             COMMAND           CREATED          STATUS           PORTS          NAMES
23aad069902a        b58dbc12f077   "/bin/bash"       13 minutes ago   Up 13 minutes   0.0.0.0:46150->8080/tcp   crazy_goldwasser

+ project2 git:(master) X curl -X GET http://localhost:46150/user
[{"id":1,"name":"vishal","age":25,"department":"cs","subject":"devops"}, {"id":2,"name":"someone","age":25,"department":"maths","subject":"maths"}]
+ project2 git:(master) X _
```

5) Postman POST method:



The latest version of Postman is out with huge improvements! Update the app for a better experience. [Download](#)

Builder Team Library

localhost:46150/user/

POST localhost:46150/user/

Authorization Headers (1) Body Pre-request Script Tests

form-data x-www-form-urlencoded raw binary JSON (application/json)

Body

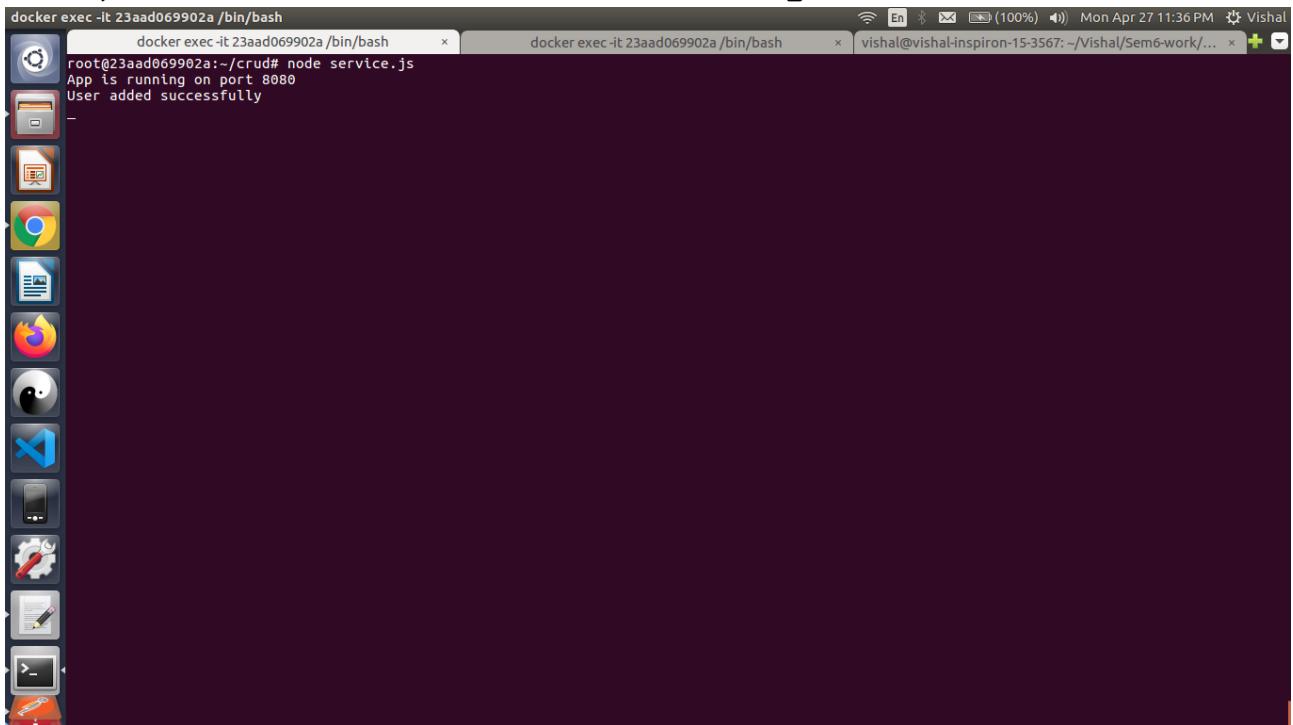
```
1 {"id":3,"name":"okane","age":25,"department":"history","subject":"history"}
```

Status: 200 OK Time: 94 ms

Pretty Raw Preview Text

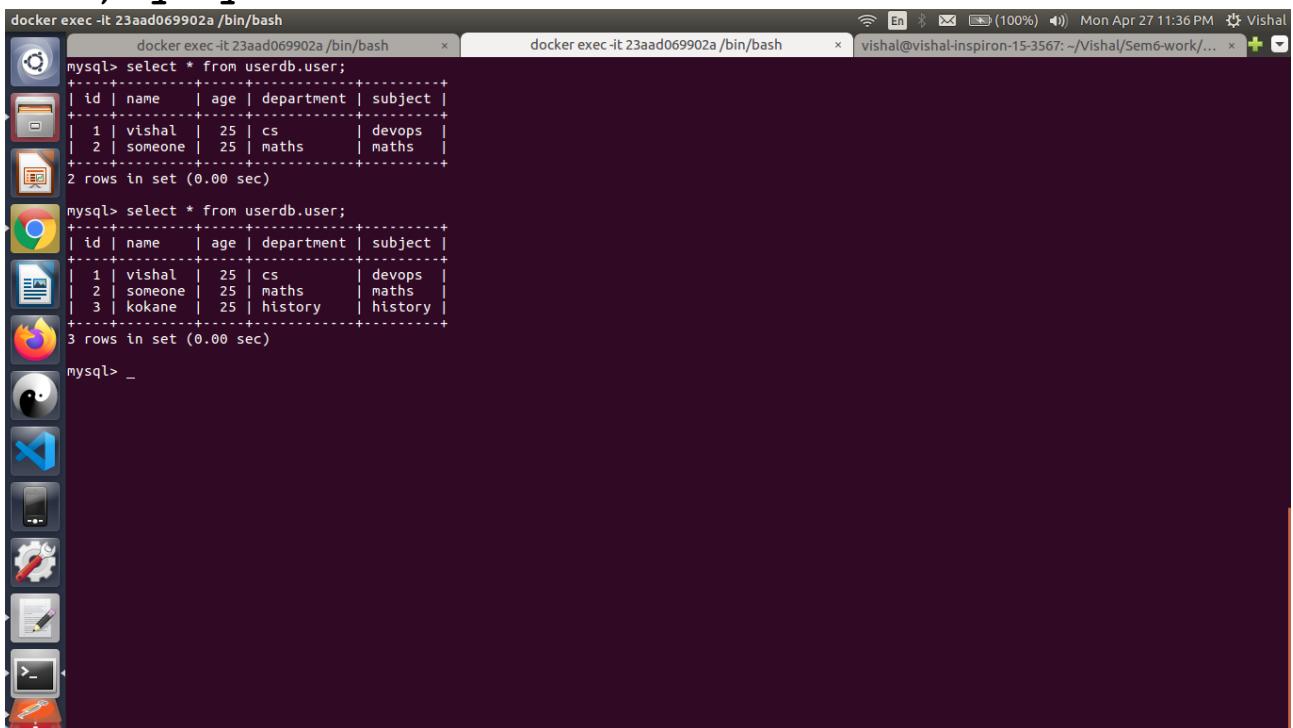
```
1 {"fieldCount":0,"affectedRows":1,"insertId":0,"serverStatus":2,"warningCount":0,"message":"","protocol41":true,"changedRows":0}
```

5.1) User added Successfully



```
root@23aad069902a:~/crud# node service.js
App is running on port 8080
User added successfully
```

5.2) Mysql result of POST



```
mysql> select * from userdb.user;
+----+-----+-----+-----+
| id | name | age | department | subject |
+----+-----+-----+-----+
| 1 | vishal | 25 | cs | devops |
| 2 | someone | 25 | maths | maths |
+----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> select * from userdb.user;
+----+-----+-----+-----+
| id | name | age | department | subject |
+----+-----+-----+-----+
| 1 | vishal | 25 | cs | devops |
| 2 | someone | 25 | maths | maths |
| 3 | kokane | 25 | history | history |
+----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> _
```

6) Postman delete data with user id =1

The screenshot shows the Postman application interface. In the left sidebar, under 'History', there is a list of requests including a recent 'DELETE' request to 'localhost:46150/user/1'. The main workspace shows a 'DELETE' request to 'localhost:46150/user/1' with the body set to JSON (application/json). The response status is 200 OK with a time of 508 ms. The response body is a JSON object:

```
1 {  
2   "error": false,  
3   "data": {  
4     "fieldCount": 0,  
5     "affectedRows": 1,  
6     "insertId": 0,  
7     "serverStatus": 34,  
8     "warningCount": 0,  
9     "message": "",  
10    "protocal41": true,  
11    "changedRows": 0  
12  },  
13  "message": "User deleted successfully."  
14 }
```

6.2) Mysql status

The screenshot shows a terminal window with three MySQL sessions. The first session shows the result of a SELECT query on the 'userdb.user' table. The second session shows the result of another SELECT query on the same table. The third session shows the result of a third SELECT query on the same table. All three sessions show the same data:

id	name	age	department	subject
1	vishal	25	cs	devops
2	someone	25	maths	maths
3	kokane	25	history	history

Each session ends with a '2 rows in set (0.00 sec)' message. The final command in each session is 'mysql> _'.

7) Postman PUT :update user name using userid

The screenshot shows the Postman application interface. In the top navigation bar, there are tabs for NEW, Runner, Import, and a collection named 'History'. The main workspace shows a 'Builder' tab with a 'PUT' method selected. The URL is set to 'localhost:46150/user/'. The 'Body' tab is active, showing raw JSON data:

```
{"id":2,"name":"vishal","age":25,"department":"maths","subject":"maths"}
```

. Below the body, the 'Headers' tab shows a Content-Type of 'application/json'. The 'Tests' tab is also visible. On the left sidebar, there's a 'History' section listing various API requests (PUT, DEL, POST, GET) made to the 'localhost:46150/user/' endpoint. The bottom right corner displays the status '200 OK' and time '68 ms'.

7.2)Mysql update status

The screenshot shows a terminal window with three tabs, all running under 'docker exec -it 23aad069902a /bin/bash'. The first tab shows the result of a 'select * from userdb.user;' query, which returns two rows: one for 'vishal' and one for 'someone'. The second tab shows the result of another 'select * from userdb.user;' query, which now includes a third row for 'kokane'. The third tab shows the result of a third 'select * from userdb.user;' query, which includes the updated row for 'kokane' and the previous two rows. All queries were run in less than 0.00 seconds.

```
mysql> select * from userdb.user;
+----+-----+-----+-----+-----+
| id | name | age | department | subject |
+----+-----+-----+-----+-----+
| 1 | vishal | 25 | cs | devops |
| 2 | someone | 25 | maths | maths |
+----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> select * from userdb.user;
+----+-----+-----+-----+-----+
| id | name | age | department | subject |
+----+-----+-----+-----+-----+
| 1 | vishal | 25 | cs | devops |
| 2 | someone | 25 | maths | maths |
| 3 | kokane | 25 | history | history |
+----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> select * from userdb.user;
+----+-----+-----+-----+-----+
| id | name | age | department | subject |
+----+-----+-----+-----+-----+
| 2 | someone | 25 | maths | maths |
| 3 | kokane | 25 | history | history |
+----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> select * from userdb.user;
+----+-----+-----+-----+-----+
| id | name | age | department | subject |
+----+-----+-----+-----+-----+
| 2 | vishal | 25 | maths | maths |
| 3 | kokane | 25 | history | history |
+----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

8) Postman get all records

The screenshot shows the Postman application interface. In the top navigation bar, there are buttons for NEW, Runner, Import, and a search bar. The main workspace has a 'Builder' tab selected. The URL field contains 'localhost:46150/user/'. The method dropdown is set to 'GET'. The 'Headers' tab shows '(1)' header entries. The 'Body' tab is selected, showing the response body which is a JSON array:

```
[{"id":1,"name":"vishal","age":25,"department":"cs","subject":"devops"}, {"id":2,"name":"someone","age":25,"department":"maths","subject":"maths"}]
```

. The status bar at the bottom right indicates 'Status: 200 OK' and 'Time: 37 ms'.

9) Postman delete fail record not exist

The screenshot shows the Postman application interface. In the top navigation bar, there are buttons for NEW, Runner, Import, and a search bar. The main workspace has a 'Builder' tab selected. The URL field contains 'localhost:46150/user/13'. The method dropdown is set to 'DELETE'. The 'Headers' tab shows '(1)' header entries. The 'Body' tab is selected, showing the response body which is a JSON object:

```
{ "error": false, "data": { "fieldCount": 0, "affectedRows": 0, "insertId": 0, "serverStatus": 34, "warningCount": 0, "message": "", "protocal41": true, "changedRows": 0 }, "message": "User deleted successfully." }
```

. The status bar at the bottom right indicates 'Status: 200 OK' and 'Time: 43 ms'.

10) Postman delete Success

The screenshot shows the Postman application interface. In the left sidebar, there is a history of requests. The main panel shows a DELETE request to `localhost:46150/user/1`. The response status is `200 OK` with a time of `615 ms`. The response body is a JSON object:

```
1 < [ 2   "error": false, 3     "data": { 4       "fieldCount": 0, 5       "affectedRows": 1, 6       "insertId": 0, 7       "serverStatus": 34, 8       "warningCount": 0, 9       "message": "", 10      "protocal41": true, 11      "changedRows": 0 12    }, 13    "message": "User deleted successfully." 14  ]
```

11) Postman Get by id

The screenshot shows the Postman application interface. In the left sidebar, there is a history of requests. The main panel shows a GET request to `localhost:46150/user/3`. The response status is `200 OK` with a time of `72 ms`. The response body is a JSON array:

```
1 [ {"id":3,"name":"kokane","age":25,"department":"history","subject":"history"} ]
```